CREATE IBM DB2 AND CONNECT WITH PYTHON

Team id : PNT2022TMID34246

Project name: Inventory Management system for Retailers

Connect to Db2 database on Cloud using Python

Estimated time needed: **15** minutes

Objectives

After completing this lab you will be able to:

- Import the ibm_db Python library
- Enter the database connection credentials
- Create the database connection
- Close the database connection

Import the ibm_db Python library

The ibm_db API provides a variety of useful Python functions for accessing and manipulating data in an IBM® data server database, including functions for connecting to a database, preparing and issuing SQL statements, fetching rows from result sets, calling stored procedures, committing and rolling back transactions, handling errors, and retrieving metadata.

We first import the ibm_db library into our Python Application

Execute the following cell by clicking within it and then press Shift and Enter keys simultaneously

In [1]:

```
!pip install --force-reinstall ibm_db==3.1.0 ibm_db_sa==0.3.7
```

```
Collecting ibm_db==3.1.0
  Using cached ibm_db-3.1.0-cp37-cp37m-linux_x86_64.whl
```

```
Collecting ibm_db_sa==0.3.7
  Downloading ibm_db_sa-0.3.7.tar.gz (30 kB)
  Preparing metadata (setup.py) ... done
Collecting sqlalchemy>=0.7.3
  Downloading SQLAlchemy-1.4.27-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x8
6_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.6 MB)
     |████████████████████████████████| 1.6 MB 12.0 MB/s
Collecting importlib-metadata
  Downloading importlib_metadata-4.8.2-py3-none-any.whl (17 kB)
Collecting greenlet!=0.4.17
  Downloading greenlet-1.1.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x
86_64.whl (150 kB)
     |████████████████████████████████| 150 kB 91.5 MB/s
Collecting typing-extensions>=3.6.4
  Downloading typing_extensions-4.0.1-py3-none-any.whl (22 kB)
Collecting zipp>=0.5
  Downloading zipp-3.6.0-py3-none-any.whl (5.3 kB)
Building wheels for collected packages: ibm-db-sa
  Building wheel for ibm-db-sa (setup.py) ... done
  Created wheel for ibm-db-sa: filename=ibm_db_sa-0.3.7-py3-none-any.whl size
=29318 sha256=92ade399444668b3672266d2df3f1b8d2c61bd679fd2a9cde25af492eeb4498
4
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/4a/e9/e7/0ee334a6cb
2f09ec45978e05837b66f59494b482ef38c7ae33
Successfully built ibm-db-sa
Installing collected packages: zipp, typing-extensions, importlib-metadata, g
reenlet, sqlalchemy, ibm-db, ibm-db-sa
  Attempting uninstall: zipp
    Found existing installation: zipp 3.6.0
    Uninstalling zipp-3.6.0:
      Successfully uninstalled zipp-3.6.0
  Attempting uninstall: typing-extensions
    Found existing installation: typing-extensions 4.0.1
    Uninstalling typing-extensions-4.0.1:
      Successfully uninstalled typing-extensions-4.0.1
  Attempting uninstall: importlib-metadata
    Found existing installation: importlib-metadata 4.8.2
    Uninstalling importlib-metadata-4.8.2:
      Successfully uninstalled importlib-metadata-4.8.2
  Attempting uninstall: sqlalchemy
    Found existing installation: SQLAlchemy 1.3.24
    Uninstalling SQLAlchemy-1.3.24:
      Successfully uninstalled SQLAlchemy-1.3.24
  Attempting uninstall: ibm-db
    Found existing installation: ibm-db 3.1.0
    Uninstalling ibm-db-3.1.0:
      Successfully uninstalled ibm-db-3.1.0
  Attempting uninstall: ibm-db-sa
    Found existing installation: ibm-db-sa 0.3.3
    Uninstalling ibm-db-sa-0.3.3:
      Successfully uninstalled ibm-db-sa-0.3.3
Successfully installed greenlet-1.1.2 ibm-db-3.1.0 ibm-db-sa-0.3.7 importlib-
metadata-4.8.2 sqlalchemy-1.4.27 typing-extensions-4.0.1 zipp-3.6.0
```

In [2]:

```
import ibm_db
```

When the command above completes, the ibm_db library is loaded in your notebook.

Identify the database connection credentials

Connecting to dashDB or DB2 database requires the following information:

- Driver Name
- Database name
- Host DNS name or IP address
- Host port
- Connection protocol
- User ID (or username)
- User Password

Notice: To obtain credentials please refer to the instructions given in the first Lab of this course

Now enter your database credentials below and execute the cell with Shift + Enter

In [3]:

```
#Replace the placeholder values with your actual Db2 hostname, username, and
password:
dsn_hostname = "2d46b6b4-cbf6-40eb-bbce-
6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud" # e.g.:
"54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud"
dsn_uid = "vjd29721"        # e.g. "abc12345"
dsn_pwd = "6TTgx8MRBzT45o3q"      # e.g. "7dBZ3wWt9XN6$o0J"

dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "BLUDB"            # e.g. "BLUDB"
dsn_port = "32328"                # e.g. "32733"
dsn_protocol = "TCPIP"            # i.e. "TCPIP"
dsn_security = "SSL"              #i.e. "SSL"
```

Create the DB2 database connection

Ibm_db API uses the IBM Data Server Driver for ODBC and CLI APIs to connect to IBM DB2 and Informix.

Lets build the dsn connection string using the credentials you entered above

In [4]:

```
#DO NOT MODIFY THIS CELL. Just RUN it with Shift + Enter
#Create the dsn connection string
dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY={7};").format(dsn_driver, dsn_database, dsn_hostname, 
dsn_port, dsn_protocol, dsn_uid, dsn_pwd,dsn_security)

#print the connection string to check correct values are specified
print(dsn)
```

```
DRIVER={IBM DB2 ODBC DRIVER};DATABASE=BLUDB;HOSTNAME=2d46b6b4-cbf6-40eb-bbce-
6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32328;PROTOC
OL=TCPIP;UID=vjd29721;PWD=6TTgx8MRBzT45o3q;SECURITY=SSL;
```

Now establish the connection to the database

In [5]:

```
#DO NOT MODIFY THIS CELL. Just RUN it with Shift + Enter
#Create database connection

try:
    conn = ibm_db.connect(dsn, "", "")
    print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, 
"on host: ", dsn_hostname)

except:
    print ("Unable to connect: ", ibm_db.conn_errormsg() )
```

```
Connected to database:  BLUDB as user:  vjd29721 on host:  2d46b6b4-cbf6-40eb
-bbce-6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud
```

Congratulations if you were able to connect successfuly. Otherwise check the error and try again.

In [6]:

```
#Retrieve Metadata for the Database Server
server = ibm_db.server_info(conn)

print ("DBMS_NAME: ", server.DBMS_NAME)
print ("DBMS_VER:  ", server.DBMS_VER)
print ("DB_NAME:   ", server.DB_NAME)
```

```
DBMS_NAME:  DB2/LINUXX8664
DBMS_VER:   11.05.0600
DB_NAME:    BLUDB
```

In [7]:

```
#Retrieve Metadata for the Database Client / Driver
client = ibm_db.client_info(conn)

print ("DRIVER_NAME:          ", client.DRIVER_NAME)
print ("DRIVER_VER:           ", client.DRIVER_VER)
print ("DATA_SOURCE_NAME:     ", client.DATA_SOURCE_NAME)
print ("DRIVER_ODBC_VER:      ", client.DRIVER_ODBC_VER)
print ("ODBC_VER:             ", client.ODBC_VER)
print ("ODBC_SQL_CONFORMANCE: ", client.ODBC_SQL_CONFORMANCE)
print ("APPL_CODEPAGE:        ", client.APPL_CODEPAGE)
print ("CONN_CODEPAGE:        ", client.CONN_CODEPAGE)
```

```
DRIVER_NAME:           libdb2.a
```

```
DRIVER_VER:             11.05.0500
DATA_SOURCE_NAME:       BLUDB
DRIVER_ODBC_VER:        03.51
ODBC_VER:               03.01.0000
ODBC_SQL_CONFORMANCE:   EXTENDED
APPL_CODEPAGE:          1208
CONN_CODEPAGE:          1208
```

Close the Connection

We free all resources by closing the connection. Remember that it is always important to close connections so that we can avoid unused connections taking up resources.

In [8]:

```
ibm_db.close(conn)
```

Out[8]:

```
True
```